Detailed Action

Claims 47-108 have been examined.

Claims 52-53, 71-72 and 93-94 have been objected to as containing allowable subject matter, yet dependant upon rejected base claims.

Claims 47-51, 54-70, 73-92 and 95-109 have been rejected.

Claim Objections

1. Claim 47 line 8 should not have a comma.

Claim 55 line 3 should read "memory location area stored".

Claim 58 line 8 should read "a failure subsequentially occurs" because "a subsequent failure" implies a previous failure.

Claim 60 lines 8-9 should move the comma to after the "and" to read "to the checkpoint and, for each duplicate".

Claim 62 line 5 should move the comma to after the "and" to read "for the process and, in response".

Claim 76, the last line should move the comma to after the "and" to read "node and, for each".

Allowable Subject Matter

2. Claims 52-53, 71-72 and 93-94 are objected to as containing allowable subject matter while being dependent on rejected base claims.

Within claims 52, 71 and 93, within each claim as a whole, the examiner deems the novel limitation to be that the process stores the contents of a memory location from the first set earlier than it would otherwise be stored when the processor needs to modify the memory location during the second checkpoint interval.

Claim Rejections - 35 USC § 112

The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

3. Claim 50 is rejected under 35 U.S.C. 112 first paragraph as lacking enablement within the specification. Claim 50 lines 1-2 states that the processor is configured to enter a barrier. This is not completely accurate. According to pages 5 and 27 of the specification, it is the processes that enter the barrier. The processor is a physical component and can not be said to actually do anything except execute instructions.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 47-51, 54-57, 66-70, 73-76, 88-92 and 95-98 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chandy (US Patent 6,898,791) in view of Fleming (US Patent 6,023,772).

5. As per claim 47, Chandy ('791) discloses a compute node capable of operating as part of a distributed system, comprising:

memory (column 27 lines 50-57); and

a processor configured to access the memory to perform a process in a distributed computation running on the distributed system (column 27 lines 50-57).

Chandy ('791) does not expressly disclose that recording the second state is performed while creating the checkpoint from the first state, or that the tasks are performed by the same processor.

Chandy ('791) teaches that each component archives its own state (column 22 lines 10-18). It would be obvious then that, when each component in a distributed system archives its own state, the various components may archive simultaneously or one may archive its state while another is executing as the execution of one component is not synchronized with another.

Chandy ('791) also teaches that multiple components of a distributed system may reside on the same computer (column 4 lines 24-27). It would also be obvious then that the

archiving and executing of components be performed by the same processor when each exists as a process on a multi-tasking computer.

Chandy ('791) does not expressly disclose the distributed system wherein the saved states are a set of memory locations modified by the processor during a checkpoint interval.

Fleming ('772) teaches a duplicate processor system wherein the information used for backup purposes is an update log (see abstract) of changes to memory (column 2 lines 6-10).

At the time of invention it would have been obvious to a person of ordinary skill in the art to modify the distributed state archiving taught by Chandy ('791) such that the archiving states include a log of modified memory locations. This modification would have been obvious because a memory snapshot allows for a process to be restored on failure (Fleming ('772) column 1 line 63 through column 2 lines 5). Additionally, Chandy ('791) mentions that a state should be saved as incremental changes (column 24 lines 31-34), implying that memory changes are saved as process events can not be incrementally saved.

6. As per claim 48, Chandy ('791) in view of Fleming ('772) discloses the compute node of claim 47 wherein the processor is further configured to write protect the first set

of memory locations (see abstract, a freeze method is used before the state is saved) before modifying the second set of memory locations (as described for claim 47, as each distributed component is a process on a multi-tasking processor (Chandy ('791) column 4 lines 24-27), it is obvious that a first process archives its state before the second process executes a modification).

7. As per claim 49, Chandy ('791) in view of Fleming ('772) discloses the compute node of claim 48 wherein the processor is further configured to suspend the process between the first and second checkpoint intervals (see abstract of Chandy ('791), a freeze method is entered before saving the state), the processor being further configured to write protect the first set of memory locations while the process is suspended (Chandy ('791) column 8 lines 21-30, the frozen process retains its persistent store. The examiner takes official notice that it is well known for an operating system on a multi-tasking system to protect the memory space of various suspended processes so they are not altered by another process).

8. As per claim 50, Chandy ('791) in view of Fleming ('772) discloses the compute node of claim 49 wherein the processor is further configured to enter a barrier following the completion of write protecting the first set of memory locations (see Chandy ('791) abstract, the state is archived after the freeze method is run) and exit the barrier before resuming the process during the second checkpoint interval (see abstract of Chandy ('791) a thaw method is run after the state is archived).

9. As per claim 51, Chandy ('791) in view of Fleming ('772) discloses the compute node of claim 48 wherein the processor is further configured to create the checkpoint by storing the contents of the first set of memory locations (Chandy ('791) column 24 lines 57-61), the processor being further configured to remove the write protection for a memory location from the first set when the processor needs to modify the memory location during the second checkpoint interval after the contents of the memory location has been stored (see Chandy ('791) abstract, after saving the state a process is thawed and resumes execution).

10. As per claim 54, Chandy ('791) in view of Fleming ('772) discloses the compute node of claim 47 further comprising a checkpoint file, wherein the processor is further configured to create the checkpoint by storing the contents of the first set of memory locations to the checkpoint file (Chandy ('791) column 8 lines 17-20).

11. As per claim 55, Chandy ('791) in view of Fleming ('772) discloses the compute node of claim 54 wherein the processor is further configured to remove the record of a memory location from the first set after the contents from the memory location is stored in the checkpoint file (Chandy ('791) column 8 lines 28-30).

12. As per claim 56, Chandy ('791) in view of Fleming ('772) discloses the compute node of claim 47 wherein the processor is further

configured to configured to create the checkpoint by storing the contents of the first set of memory locations to non-volatile storage (Chandy ('791) column 8 lines 17-20. Additionally the examiner takes official notice that it is obvious to archive the distributed process states to non-volatile storage because non-volatile storage allows for archives to be stored without power).

13. As per claim 57, Chandy ('791) in view of Fleming ('772) discloses the compute node of claim 47 wherein the processor is further
configured to

store in the memory a copy of each message output from the compute node during the process until an acknowledgement is received (Chandy ('791) column 25 lines 42-45), and

output each message copied in the memory that does not receive an acknowledgement (Fleming ('772) column 4 lines 51-58), and

wherein the processor is further configured to receive messages during the process, and output an acknowledgement for each message received (Chandy ('791) column 25 lines 23-25),

the processor being further configured to recognize and discard duplicate messages received by the compute node (Fleming ('772) column 10 lines 25-35), and for each duplicate message, output an acknowledgement (Fleming ('772) column 5 lines 14-21).

14. As per claims 66-70 and 88-92, these sets of claims recite limitations found in claims 47-51 and are rejected on the same grounds as claims 47-51.

15. As per claims 73-76 and 95-98, these sets of claims recite limitations found in claims 54-57 and are rejected on the same grounds as claims 54-57.

16. Claims 58-61, 77-83, 99-102 and 107-109 are rejected under 35 U.S.C. 103(a) as being unpatentable over Fleming ('772) in view of Chandy ('791).

17. As per claim 58, Fleming ('772) discloses a compute node comprising:

memory (Figure 2, each node has state change buffer, input queues, and resend queues in addition to the application memory); and

a processor configured to perform a process (column 3 lines 16-17),

the processor being further configured to store in the memory a copy of each message output from the compute node until an acknowledgement is received (column 5 lines 14-28, a messages are removed from the resend queue after they are acknowledged),

the processor being further configured to create a checkpoint, and if a subsequent failure occurs, roll the compute node back to the checkpoint and output each message copied in the memory that does not receive an acknowledgement after the compute node is rolled back to the checkpoint (column 4 lines 51-60).

Fleming ('772) does not expressly disclose the system wherein the node is part of a distributed system.

Chandy ('791) teaches a distributed system that performs checkpointing (see abstract).

At the time of invention it would have been obvious to a person of ordinary skill in the art to modify the process logging system of Fleming ('772) such that the process is part of a distributed process. This modification would have been obvious because distributed systems allow concurrent computations to use distributed resources (Chandy ('791) column 1 lines 45-58).

18. As per claim 59, Fleming ('772) in view of Chandy ('791) discloses The compute node of claim 58 wherein the processor is further configured to receive messages, and output an acknowledgement for each message received (Fleming ('772) column 9, lines 21-26)

the processor being further configured to recognize and discard duplicate messages received by the compute node (Fleming ('772) column 10 lines 32-35), and for each duplicate message, output an acknowledgement (Fleming ('772) column 10 lines 21-26).

19. As per claim 60, Fleming ('772) discloses a compute node, comprising:

a processor configured to perform a process (column 3 lines 16-17),

the processor being further configured to receive messages, and output an acknowledgement for each message received (Fleming ('772) column 9, lines 21-26),

the processor being further configured to create a checkpoint, and if a subsequent failure occurs, roll the compute node back to the checkpoint (column 4 lines 51-60),

recognize and discard duplicate messages received by the compute node after the compute node is rolled back to the checkpoint (column 10 lines 32-35) and, for each duplicate message, output an acknowledgement (column 10 lines 21-26).

Fleming ('772) does not expressly disclose the system wherein the node is part of a distributed system.

Chandy ('791) teaches a distributed system that performs checkpointing (see abstract).

At the time of invention it would have been obvious to a person of ordinary skill in the art to modify the process logging system of Fleming ('772) such that the process is part of a distributed process. This modification would have been obvious because distributed systems allow concurrent computations to use distributed resources (Chandy ('791) column 1 lines 45-58).

20. As per claim 61, Fleming ('772) in view of Chandy ('791) discloses the compute node of claim 60 further comprising memory,

wherein the processor is further configured to store in the memory a copy of each message output from the compute node until an acknowledgement is received (column 5 lines 14-28, a messages are removed from the resend queue after they are acknowledged), the processor being further configured to output each message copied in the memory that does not receive an acknowledgement (column 4 lines 51-60).

21. As per claims 77-80 and 99-102, these claim sets recite limitations respectively found in claims 58-61 and are respectively rejected on the same grounds as claims 58-61.

22. As per claim 81, Fleming ('772) discloses computer readable media embodying a program of instructions comprising a checkpoint library executable by a processor having access to an operating system to perform a process in, the checkpoint library comprising:

instructions to create a checkpoint for the process (see abstract, the primary transmits status logs to the secondary),

the creation of the checkpoint being transparent to the operating system (as shown in Figure 2, the fault-tolerant services layer 23 is above the operating system 22) and the distributed application (as shown in Figure 2, the fault-tolerant service layer 23 is below the application).

Fleming ('772) does not expressly disclose the system wherein the application is part of a distributed application running on a distributed system.

Chandy ('791) teaches a distributed system that performs checkpointing (see abstract).

At the time of invention it would have been obvious to a person of ordinary skill in the art to modify the process logging system of Fleming ('772) such that the process is part of a distributed process. This modification would have been obvious because distributed systems allow concurrent computations to use distributed resources (Chandy ('791) column 1 lines 45-58).

23. As per claim 82, Fleming ('772) in view of Chandy ('791) discloses the computer readable media of claim 81 wherein the instructions
to create the checkpoint comprise

instructions to record a first set of memory locations modified by the process during a first checkpoint interval (Fleming ('772) column 3 lines 29-34, while memory locations are not explicitly disclosed, it is obvious that modified memory locations is part of the state of a process), and

instructions to create the checkpoint from the contents of the first set of memory locations (column 3 lines 35-38), while recording a second set of memory locations modified by the process during a second checkpoint interval (as stated in the abstract, the checkpointing of each primary process is performed by its partner processor, this

allows the checkpointing to occur while the original process continues to execute).

24. As per claim 83, Fleming ('772) in view of Chandy ('791) the computer readable media of claim 81 wherein

the process is performed in a compute node in the distributed system, and wherein

the instructions to create the checkpoint comprise instructions to store in memory a copy of each message output from the compute node during the process until an acknowledgement is received (Fleming ('772) column 5 lines 15-21), and output each message copied in the memory that does not receive an acknowledgement (Fleming ('772) column 4 lines 51-58), and

instructions to receive messages during the process, output an acknowledgement for each message received, recognize and discard duplicate messages received by the compute node (Fleming ('772) column 10 lines 25-35), and for each duplicate message, output an acknowledgement (Fleming ('772) column 5 lines 14-21).

25. As per claim 107, Fleming ('772) discloses a method of migrating a process from a first compute node to a second compute node, each of the first and second compute nodes having an operating system, the method comprising:

creating a checkpoint for the process in the first compute node (Figure 3); and

migrating the process to the second compute node by providing the second

compute node with the checkpoint (as shown in Figured 3 and 4, the process migrates when a failure occurs) without migrating the operating system from the first compute node to the second compute node (no mention is made of migrating the operating system, additionally, Figure 2 shows the operating system level 22 being below the fault-tolerant service layer 23).

Fleming ('772) does not expressly disclose the system wherein the application is part of a distributed application running on a distributed system.

Chandy ('791) teaches a distributed system that performs checkpointing (see abstract).

At the time of invention it would have been obvious to a person of ordinary skill in the art to modify the process logging system of Fleming ('772) such that the process is part of a distributed process. This modification would have been obvious because distributed systems allow concurrent computations to use distributed resources (Chandy ('791) column 1 lines 45-58).

26. As per claims 108 and 109, these claims recite limitations found in claims 82 and 83, respectively, and are respectively rejected on the same grounds as claims 82 and 83.

Claims 62-65, 84-87 and 103-106 are rejected under 35 U.S.C. 103(a) as being

unpatentable over Fleming (‘772) in view of Chandy (‘791) and Sun (US Patent 7,065,549).

27. As per claim 62, Fleming ('772) discloses a compute node, comprising:

a processor configured to perform a process (column 3 lines 16-17),

the processor being further configured to create a checkpoint for the process (column 4 lines 51-60), and

store the checkpoint to non-volatile memory (column 2 lines 17-20).

Fleming ('772) does not expressly disclose the system wherein the node is part of a distributed system.

Chandy ('791) teaches a distributed system that performs checkpointing (see abstract).

At the time of invention it would have been obvious to a person of ordinary skill in the art to modify the process logging system of Fleming ('772) such that the process is part of a distributed process. This modification would have been obvious because distributed systems allow concurrent computations to use distributed resources (Chandy ('791) column 1 lines 45-58).

Fleming ('772) does not expressly disclose the system wherein the storing of the checkpoint is in response to a preemptive scheduling request which results in halting the process.

Sun ('549) teaches a distributed system with process migration between nodes (see abstract). Within the system a user issues a migration request which is then sent to a scheduler (column 4 lines 52-56 and as shown in Figure 2) which then collects messages including the process state (Figure 2).

At the time of invention it would have been obvious to a person of ordinary skill in the art to modify the process logging system disclosed by Fleming ('772) in view of Chandy ('791) such that a user can request a process migration which then triggers a preemptive scheduling request, as taught by Sun ('549). This modification would have been obvious because allowing a user-initiated process migration allows for a process to be ported from one computer platform to an upgraded one (Sun ('549) column 2 lines 9-19).

28. As per claim 63, Fleming ('772) in view of Chandy ('791) and Sun ('549) discloses the compute node of claim 62 wherein the processor is further configured to use the stored checkpoint to resume the process (as shown in Sun ('549) Figure 2).

29. As per claim 64, Fleming ('772) in view of Chandy ('791) and Sun ('549) discloses the compute node of claim 62 wherein the processor is further configured to perform a process in a second distributed computation running on the distributed system after the previous process is halted (Sun ('549) column 1 lines 56-57)).

30. As per claim 65, Fleming ('772) in view of Chandy ('791) and Sun ('549) discloses the compute node of claim 62 wherein the processor is further configured to perform a second process in the distributed computation previously being performed by another compute node in the distributed system, the processor being further configured to resume the second process from the checkpoint last taken by said another compute node for the second process (Sun ('549) column 2 lines 15-19, this is what process migration is. All nodes in Sun ('549) are capable of taking over and giving up processes).

31. As per claims 84-87 and 103-106, these sets of claims each recite limitations respectively found in claims 62-65 and are respectively rejected on the same grounds as claims 62-65.

Conclusion

The prior art made of record on accompanying PTO 892 form and not relied upon is considered pertinent to applicant's disclosure.

Contact Information

Any inquiry concerning this communication or earlier communications from the examiner should be directed to JOSEPH SCHELL whose telephone number is

(571)272-8186. The examiner can normally be reached on Monday through Friday 9AM-4:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Scott Baderman can be reached on (571) 272-3644. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Scott T Baderman/
Supervisory Patent Examiner, Art Unit 2114

JS